Convex Optimization For Machine Learning (cvx4ml)
Konstantin Burlachenko (bruziuz@stanford.edu, burlachenkok@gmail.com)

Datasets
For test my solvers and implementation which I created via using CVXPY and SkLearn I used following data, splitted by holdout cross-validation into 70%/30% train/dev set:
1. Regression Set-1, features from 3-14 and target variable is feature 2
   https://www.kaggle.com/harlfoxem/housesalesprediction/data
2. Classification Set-1, features 1-19, and target variable is feature 20
   https://www.kaggle.com/kiprop/projects/exoxgender/data

Results about model comparisons for this datasets

<table>
<thead>
<tr>
<th>Model name</th>
<th>Mean Square Error on Train</th>
<th>Mean Square Error on Dev</th>
</tr>
</thead>
<tbody>
<tr>
<td>linear_regression_L2</td>
<td>1792.8452355</td>
<td>2739.7680847</td>
</tr>
<tr>
<td>linear_regression_L2_with_L1regularation, lamba=0.1</td>
<td>1800.6951332</td>
<td>2741.376932</td>
</tr>
<tr>
<td>linear_regression_L2_with_L2regularation, lamba=0.2</td>
<td>1783.9151759</td>
<td>2792.9655686</td>
</tr>
<tr>
<td>linear_regression_L2_with_stochastic_robust, buckets=20</td>
<td>1783.0598205</td>
<td>2786.088867</td>
</tr>
<tr>
<td>linear_regression_L2_with_trust_region, D=2000, d=00.00</td>
<td>1620.8061679</td>
<td>2752.2135917</td>
</tr>
<tr>
<td>linear_regression_L2_with_trust_region, D=2000, d=10.00</td>
<td>1776.7521649</td>
<td>2775.8763092</td>
</tr>
<tr>
<td>linear_regression_Linf</td>
<td>4020.5919818</td>
<td>7765.1882507</td>
</tr>
<tr>
<td>Linear_intercept_L2</td>
<td>2598.2907365</td>
<td>4971.2401347</td>
</tr>
<tr>
<td>linear_regression_Huber, MHuber = 50.0</td>
<td>1800.2455136</td>
<td>2818.7349417</td>
</tr>
<tr>
<td>linear_regression_L1</td>
<td>1856.1600085</td>
<td>2819.4144265</td>
</tr>
<tr>
<td>linear_regression_L1_with_stochastic_robust, buckets=20</td>
<td>2111.7709395</td>
<td>3214.1590754</td>
</tr>
<tr>
<td>linear_regression_L1_with_trust_region, D=2000, d=00.00</td>
<td>1872.5267745</td>
<td>2889.6036294</td>
</tr>
<tr>
<td>linear_regression_DeducTime</td>
<td>1682.1477223</td>
<td>2818.4116387</td>
</tr>
</tbody>
</table>

My solvers are based on Interior-Point method. Nesterov and Nemirovski were the first to find out that interior-point methods can solve many convex optimization problems. It's not first-order. It doesn't use slow methods based on gradients or subgradients. Second order methods can be used by people with advanced Linear Algebra level, unfortunately. This is because Hessian structure should be exploited very carefully.

Covered and Improved Models from classical ML
Convex optimization is good for model fitting. But it’s not only about that logistic regression, linear regression, SVM. But convex optimization also. A taking idea of convex optimization there are a lot of freedom in and models in classical Machine Learning. Unfortunately it’s approximately immediately close the doors to closed form solutions. General schema for the data and loss function $\sum_{i=1}^{n} (\ell(x_i, y_i) + J(x_i))$ generates a lot of convex optimization problems - if $\ell(x, y)$ is convex and $L(x, y)$ is convex in $x$ for fixed $y$. I implemented solvers for several problems for classification:
1. Logistic regression, SVM, SVM without slack constraints
2. Variations of improved SVM

I showed extra things:
1. How exact constraints to linear regression can be exploited
2. How piecewise affine convex function into data
3. How improved L1 heuristic can be exploited

I implemented solvers for several problems for regression:
1. Dead zone fit and Log Barrier fit, Huber for (Robust Linear regression)
2. Fit with $L_1$, norm with trust region, stochastic $L_2$
3. Fit with $L_1$ norm (least-square), $L_2$ norm with trust region, stochastic $L_2$
4. Fit with $L_1$ norm and $L_2$ norm as regularization (Lasso regression)
5. Fit with $L_1$ norm and $L_2$ norm as regularization (Ridge regression)
6. Fit with $L_0$ (Chebyshev optimization problem), stochastic $L_0$

My solvers are based on Interior-Point method. Nesterov and Nemirovski were the first to point out that interior-point methods can solve many convex optimization problems. It’s not first-order. It doesn’t use slow methods based on gradients or subgradients. Second order methods can be used by people with advanced Linear Algebra level, unfortunately. This is because Hessian structure should be exploited very carefully.

Problem (Motivation, 5 sentence)
"Humaniy is a wandering fires in the fog. The appearance of breakthroughs through the fog from one flame to another can be called a miracle - A.N. Kolmogorov". Machine Learning connects engineering fields with usual people life. But Machine Learning can be improved by mathematical optimization, which has already become an important tool in many areas.[2]

When I visited Stanford at 2NOV2017 I heard from prof. S.Boyd that this days our world have been fractured, even several years ago all people was happy with pure convex optimization. In this work I tried to make some steps to find interesting things in Convex Optimization which is useful Machine Learning with the goal to “connect the fires”.

One example is $L_p$ norm for feature selection, instead greedy approach: E. Candès to S.Boyd: “L1 is least squares for 21 century.”[1]

Discussion
Convex optimization relative to Machine Learning:
1. Basic way create models which are convex in parameters and find parameters.
2. Introduce new concepts – stochastic, worst case hard constraints to field of ML
3. Exploiting this concepts open a way to connect ML with existing people knowledge base via pull this into ML algorithms in form of constraint or via multigoal optimization technique.
4. Give you a way combine several models sequentially – you start with one. And then formulate trust region around old solution and move to another solution.
5. Increase speed for existing models which are used in classical ML. Interior-point methods can solve optimization problem in a number of iterations that is almost alway in the range between 10 and 100. Ignoring any structure in the problem (such as sparsity), each step requires on the order of $\max(\mu^2, \sigma^2, F)$. Where:
   $F$ – cost for compute gradients and hessians, $\mu$ – number of constraints $\mu$, $\sigma$ - number of variables
6. It’s very hard to say how things are changing. But ideas of gradients, subgradients, heavy-ball method variation is already in the field of interest of Machine Learning community. Even they was born in mathematical optimization. It is less well known that convex optimization has very effective algorithms that can reliably and efficiently solve even large convex problems besides optimization community.

Future Works
1. Find applications from convex optimization for ML based on SDP, E.g. Larger-Now (minimum volume) ellipsoid around train set. (Find into active points on surface. And make heuristic decision on this points.)
2. Fit with trust region, stochastic
3. How fit piecewise affine convex function into data
4. Increase speed for existing models which are used in classical ML. Interior-point methods can solve optimization problem in a number of iterations that is almost always in the range between 10 and 100. Ignoring any structure in the problem (such as sparsity), each step requires on the order of $\max(\mu^2, \sigma^2, F).$ Where:
   $F$ – cost for compute gradients and hessians, $\mu$ – number of constraints, $\mu$, $\sigma$ - number of variables
5. Increase speed for existing models which are used in classical ML. Interior-point methods can solve optimization problem in a number of iterations that is almost always in the range between 10 and 100. Ignoring any structure in the problem (such as sparsity), each step requires on the order of $\max(\mu^2, \sigma^2, F).$ Where:
   $F$ – cost for compute gradients and hessians, $\mu$ – number of constraints, $\mu$, $\sigma$ - number of variables
6. It’s very hard to say how things are changing. But ideas of gradients, subgradients, heavy-ball method variation is already in the field of interest of Machine Learning community. Even they was born in mathematical optimization. It is less well known that convex optimization has very effective algorithms that can reliably and efficiently solve even large convex problems besides optimization community.

References
[3] "In non-convex optimziation there a a lot of room for personal expression" -S.Boyd [3]
[6] "In non-convex optimization there a a lot of room for personal expression" -S.Boyd [3]